ECE 6332 Project Proposal:

Register File Design Optimization with Virtual Prototyping Tool (ViPro)

*Group member:*

Ningxi Liu (ECE6332) Email:nl6cg@virginia.edu

Shawn Chen (ECE6332) Email:sc7cq@virginia.edu

# 1. Introduction: To realize register file optimization with ViPro

Register file (RF) is an array of processor registers in a central processing unit, and they usually have multiple read and write ports which can realize fast operation. Register file is a significant fraction of the power budget of processors, and they account for more than 25% of total power consumption in low power applications according to Nalluri’s work (2007)[[1]](#endnote-1).

ViPro is a tool designed for fast, efficient optimization and generation of memory macros and subsystems targeting diverse system level requirements across multiple process technology generations. It can estimate macro level metrics before subcomponents are completed or designed, compute the effects of a low level circuit change on the overall macro, and rapidly evaluate varying prototypes of arrays using new circuits or new cell technologies[[2]](#endnote-2).

Register file is a significant source of power consumption in high-performance processors, so the trade-off between power and delay is critical in RF design. In addition, it's common that more than 30 unique and custom register files are used in a single CPU and Soc (Eric Donkoh, DAC2012[[3]](#endnote-3)). Different topology of RF bitcell and architecture result in different power and delay, but full custom design of each RF needs lots of effort. To develop a virtual prototyping tool which can accurately model RFs' critical features without completing sub-circuits design will efficiently assist the design and optimization of RF.

# 2 Research questions

* **What’s the new characterization method for register file?**

Now there are two existing methods of characterizing a register file, which are delay and power. We need more benchmark circuits to evaluate the register file, such as yield, area and etc. Also, more parameters like process variation, technology node, transistor’s size and bank organization should be taken into account. Publications about register files design and optimization are summarized in Table 1. These strategies can be integrated into ViPro as different parameters or benchmark circuits, for example the idea showed in Ref6 that partitioning global bit-line into several local bit-lines is feasible to be included in ViPro, and so as the differential-end and single end cell structure comparison in Ref7.

Table Summary of references

|  |  |
| --- | --- |
| **Items** | **Strategy** |
| **Ref[[4]](#endnote-4)** | Analyzed the model of delay and energy by dividing register files into parts |
| **Ref[[5]](#endnote-5)** | Applied power switch to less active register cells |
| **Ref****[[6]](#endnote-6)** | Partitioning global bit-line into several local bit-lines |
| **Ref****[[7]](#endnote-7)** | Compared differential-end and single end read port structure |
| **Ref[[8]](#endnote-8)** | Utilized high threshold voltage transistor in register file |
| **Ref[[9]](#endnote-9)** | Employed yield estimation and optimization in SRAM design |

* **Which knobs are the dominant factors affecting the output generated by ViPro, and how to improve the speed of ViPro?**

Because ViPro generates output data by simulating under the sweep of each knob and calculating the delay and energy results accordingly, the simulation always takes rather a long time. However, not all of those generated results are valuable since many of them are obviously bad choices. What’s more, each knob is more effective in specific range to different application oriented register files. By generating a comprehensive output data for the first time, we can analyze the effective region of each knob, and giving some region high priority to generate results. The knowledge can be saved as feedback information for future’s simulation, by which ViPro can generate the expected results faster.

Figure 1-4 are generated by ViPro, they illustrate how the number of rows and columns affect the read & write delay and read & write energy of register files. As Figure 1 & 3 show, the number of row is more effective in deciding the delay and energy within the range of 64 to 128. Figure 2 & 4 show the number of column is more effective within the range of 1 to 4. So ViPro should choose the number of row around 64 to 128 and the number of column around 1 to 4 first in future’s simulations.

Figure Read & Write Delay depend on NROWS

Figure Read & Write Delay on NCOLS

Figure Read & Write Energy on NROWS

Figure Read & Write Energy on NCOLS

* **Can an optimization tool like ViPro generate power and delay estimates for a RF that approach the accuracy of SPICE? How to verify the model of RF in ViPro?**

ViPro is more efficient than full circuit level simulation for it utilizes a model to calculate the key feature of RF, and the simulations of ViPro are components of RF arrays which results in the inaccuracy of ViPro model. To verify the data generated by ViPro, we will compare the results of ViPro and spice simulation. There is a trade-off between the speed and accuracy of ViPro, so a balance between the two metrics needs to be reached. To be mentioned, the spice netlist of a register file array usually takes much effort to create, so it’s impossible to verify every set of results generated by ViPro.

* **How would sense amplifier (SA) affect delay and power under low voltage situation? Would it be better to have a SA or not?**

In low voltage applications, the bit-line swing fluctuates more severely because process variation has a larger influence at low voltage. SA should wait all bit-lines forms a voltage difference larger than the offset voltage, so power decreasing advantage of utilizing a SA has been weakened. On the other hand, not using a SA can reduce area overhead and decrease the power consumption by SA. By comparison the energy consumption and delay of register file with and w/o a SA in ViPro can help with making choice.

# 3. Innovations and expected output

ViPro is capable of taking process corner, technology, bit-cell topology and architecture into consideration, and knobs such as row & column number and bank size are used in estimating power consumption and delay of RF. Some other memory analyzing tools such as Cacti are only capable architecture level exploration, but they are not allowed to modify the bottom circuit level of memory. What's more, ViPro can support across technologies analyzing if there are available spice model.

Goal to develop ViPro is making it efficient and powerful software for optimization. When we finish the project, the expectation of output is a Virtual Prototyping tool which can support quick and comprehensive analysis of register file, and helps to provide an optimized register file according to specific application.

# 4.Reference

1. Rakesh Nalluri and etc. Customization of Register File Banking Architecture for Low Power. VLSID 2007. [↑](#endnote-ref-1)
2. Benton Calhoun. VIRTUAL PROTOTYPING (VIPRO) TOOL FOR MEMORY SUBSYSTEM DESIGN EXPLORATION AND OPTIMIZATION. [↑](#endnote-ref-2)
3. Eric Donkoh and etc. A hybrid and Adaptive Model for Predicting Register File and SRAM Power Using a Reference Design. DAC 2012, June 3-7, P62-67. [↑](#endnote-ref-3)
4. Kevin Linger. Automating the Design of Register Files in VLSI Chips: A Prototyping Tool. [↑](#endnote-ref-4)
5. Xuan Guan and etc. Reducing Power Consumption of Embedded Processors through Register File Partitioning and Compiler Support. ASAP 2008, P269-274. [↑](#endnote-ref-5)
6. Ataur R. Patwary. Bit-Line Organization in Register Files for Low Power AND HIGH-PERFORMANCE APPLICATIONS. ICECE 2006, P505-508. [↑](#endnote-ref-6)
7. Ting-Sheng Jau and etc. Analysis and Design of High Performance, Low Power Multiple Ports Register Files. APCCAS 2006, P1453-1456. [↑](#endnote-ref-7)
8. Hao Yan and etc. A Low-power 8-Read 4-Write Register File Design. APCPR 2010. Page 178-181. [↑](#endnote-ref-8)
9. Jim Boley and etc. Virtual Prototyper (ViPro): An SRAM Design Tool for Yield Constrained Optimization. TVLSI 2014. [↑](#endnote-ref-9)